



## FME Server API 开发示例

### 内容

- [介绍 Introduction](#)
- [使用服务器会话 Working with Server Sessions](#)
- [使用存储库 Working with Repositories](#)
- [使用工作区 Working with Workspaces](#)
- [使用资源 Working with Resources](#)
- [使用服务 Working with Services](#)
- [运行工作区 Running Workspaces](#)

### 显示代码:

- [Java](#)
- [C++](#)
- [.NET](#)

### 介绍

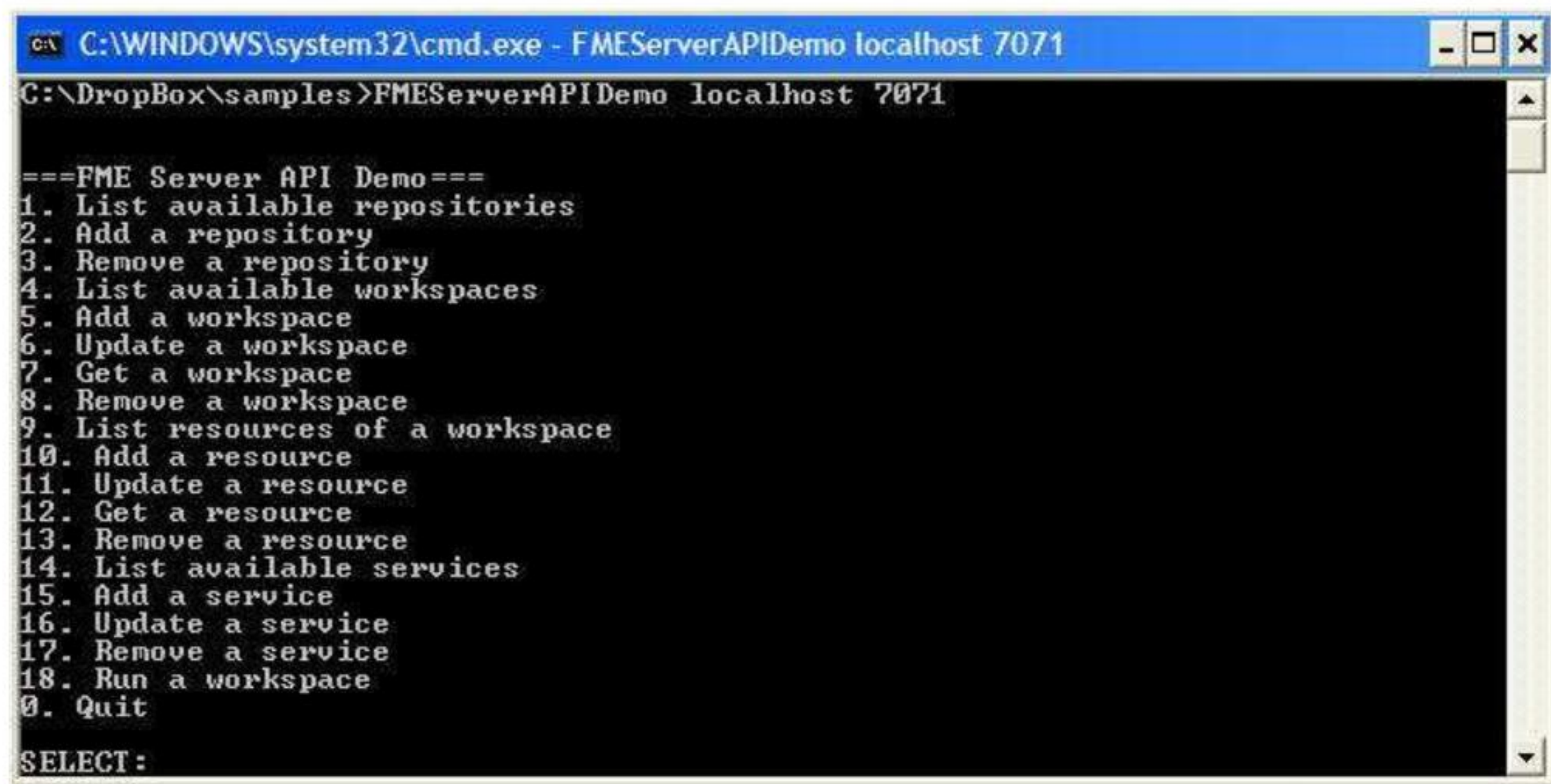
本文档是使用 FME Server API 开发应用程序的入门资料。概述了许多常见的操作，例如获取服务会话（obtaining a server session）、管理存储库项目和资源（managing repository items and resources）、运行工作区（running workspaces）。请注意，提供的示例代码旨在示例演示。

以下相关的开发链接可能会有所帮助：



[API](#)  
[Documentation](#)  
[-.NET](#)  
[API Demo Source](#)  
[File -.NET](#)

演示示例创建了一个命令行应用程序，该示例代码阐述了很多常见的 FME Server API 概念。运行示例命令行应用程序需要带两个参数。第一个参数是 FME Server 主机名，第二个参数是 FME Server 的端口号。在执行命令行应用程序时，将显示一个菜单，来提示您选择一个给定的命令。如下图。

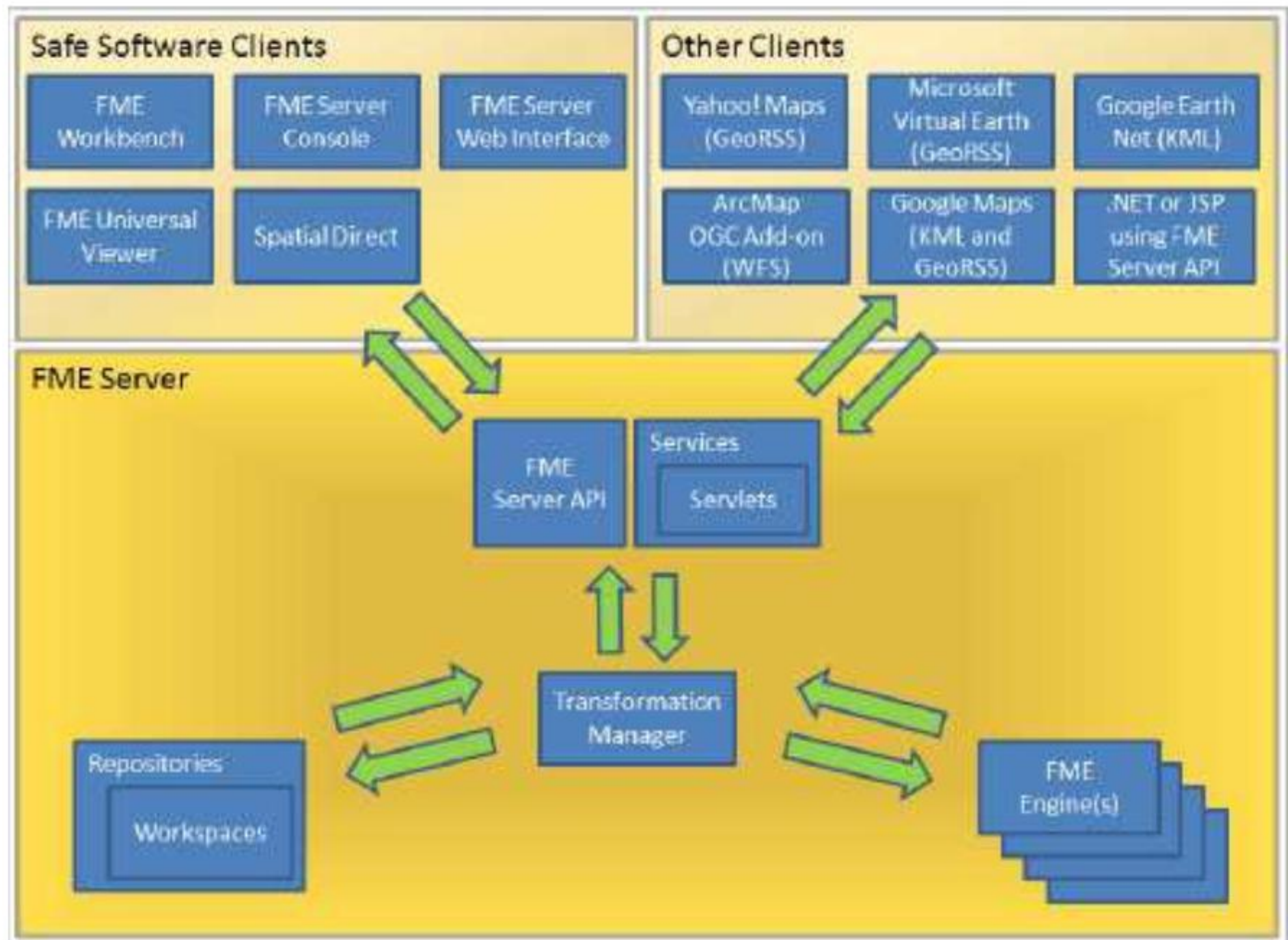


```
C:\WINDOWS\system32\cmd.exe - FMEServerAPIDemo localhost 7071
C:\DropBox\samples>FMEServerAPIDemo localhost 7071

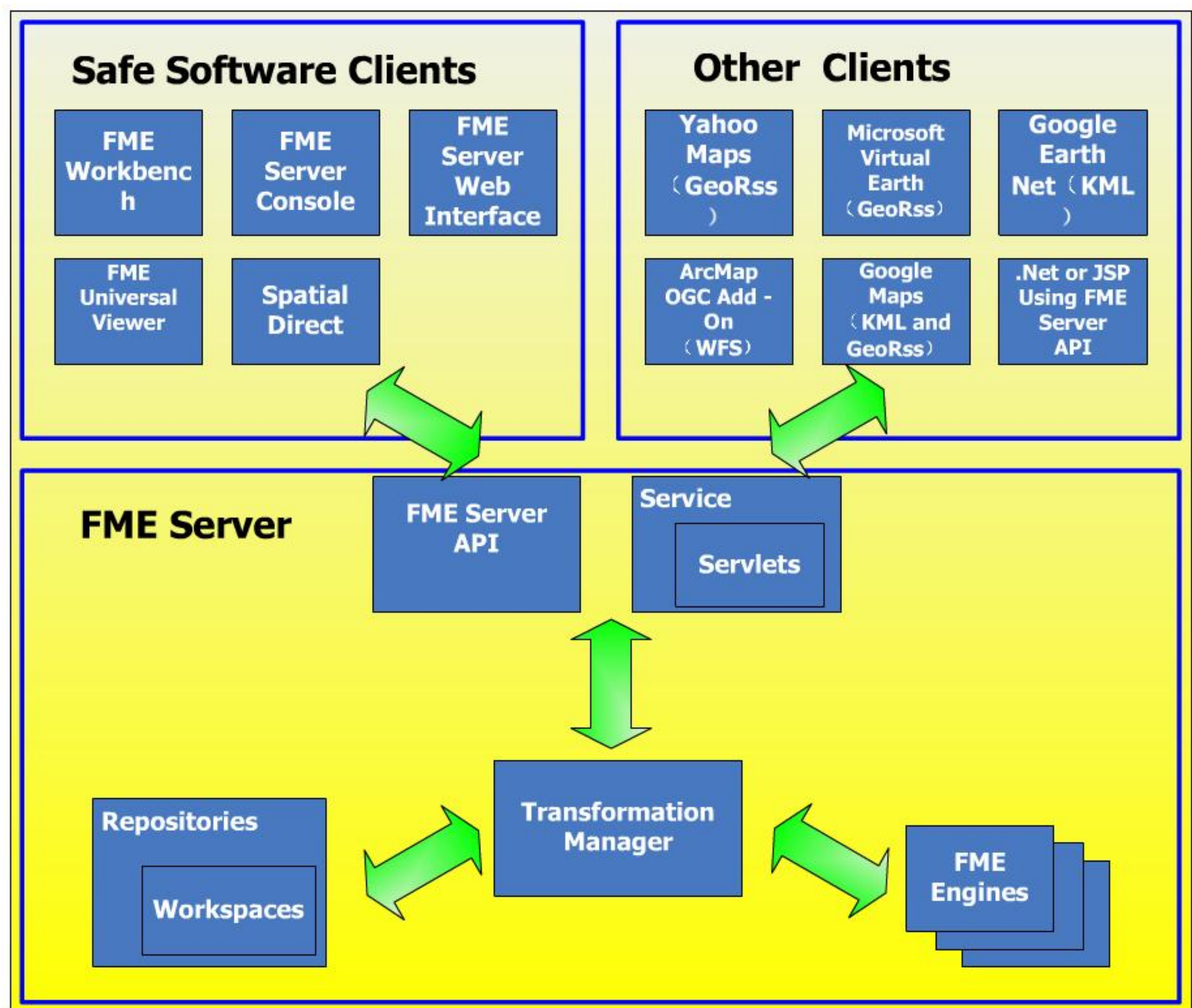
===FME Server API Demo===
1. List available repositories
2. Add a repository
3. Remove a repository
4. List available workspaces
5. Add a workspace
6. Update a workspace
7. Get a workspace
8. Remove a workspace
9. List resources of a workspace
10. Add a resource
11. Update a resource
12. Get a resource
13. Remove a resource
14. List available services
15. Add a service
16. Update a service
17. Remove a service
18. Run a workspace
0. Quit
SELECT:
```

该示例代码有 Java、c + + 和.net 三个版本。请参阅 《FME Server 开发人员指南 FME Server Developer Guide 》进一步了解有关如何设置您的开发环境的信息。









## 使用服务器会话

要使用 **FME Server API**，首先需要创建一个服务器会话。可以使用如下所示的工厂方法创建服务器会话。

```
IFMEServerSession serverSession_ =
FMEServer.CreateServerSession();
```

服务器会话一旦创建，您可以创建初始化服务器会话所需的对象。若要初始化服务器会话，需要创建连接信息对象，以指定服务器会话的连接参数。如下所示。

```
IFMEServerConnectionInfo connectionInfo_
= serverSession_.CreateServerConnectionInfo(host, port,
```



```
"" , "");  
serverSession_.Init(connectionInfo_, null);
```

服务器会话初始化后，就可以创建与 **FME Server** 进行交互的任何其他对象。包括能够管理项目和资源的存储库管理器、运行工作区（**running workspaces**）、转换管理器（**Transformation Manager**）。

使用服务器会话时特别需要注意的是确保您的服务器会话完成后得到释放。如果用户从未释放过他们的初始化服务器会话连接，您最终可能耗尽相关的连接资源。下面显示了释放服务器会话连接。

```
serverSession_.Disconnect();
```

注意： **FME Server API** 是不区分大小写的（**case insensitive**），但所有唯一标识符如存储库名称、工作区名称、资源名称、服务名称等除外。

[top](#)

## 使用存储库

存储库管理器作为 **FME Server** 的一个组件，主要用来管理存储库。存储库是一个集中存储项目、资源和启用 **FME Server** 执行其功能的其他事情的区域。一个典型的项目是一个工作区，自定义格式或自定义的转换器（**transformer**）。与项目相关联的资源。

若要访问存储库，您需要从服务器会话中获取存储库管理器对象。如下所示：

```
IFMERepositoryManager repositoryMgr_  
serverSession_.GetRepositoryManager();
```

### 如何列出可用存储库

存储库管理器对象使我们能够获得所有可用的存储库。

```
IFMERepository[] repositoryList =  
repositoryMgr_.GetRepositories(null);  
foreach (IFMERepository repository in repositoryList)  
{  
    Console.WriteLine("{0} {1}\n", repository.Name,
```



```
repository.Description);  
}
```

## 如何添加存储库

存储库由其名称唯一标识。通过指定一个新的存储库名称和存储库说明，可以创建一个新的存储库。

注意：如果您不检查存储库中是否已经存在一个具有相同名称的存储库，就添加一个存储库中具有相同的名称的存储库，您将需要处理 **FME Server** 异常或返回错误。

```
if (!repositoryMgr_.RepositoryExists(name))  
{  
    repositoryMgr_.AddRepository(name, description);  
}  
else  
{  
    throw new FMEServerException("There is already a  
repository with this name!");  
}
```

## 如何删除存储库

删除存储库需要存储库的名称。必须知道移除存储库将同时移除它包含的所有项目和资源。

```
if (!repositoryMgr_.RemoveRepository(name))  
{  
    throw new FMEServerException("There is no repository with  
this name!");  
}
```

## 使用工作区

工作区是一种类型的存储库项目，因此在这一节中的概念在其他的存储库项目如自定义格式和自定义的转换器也可用。工作区通常的文件扩展名为 **FMW**，而自定义格式文件扩展名为 **FDS**，自定义转换器文件扩展名为 **FMX**。

**FME** 工作区通过 **FME Workbench** 创建，可以发布到一个 **FME Server** 存储库中，以使多个用户可以访问它们。**FME Server** 存储库中的工作区还可以下载、修改、然后再次发布。

默认安装的 **FME Server** 中附带了许多示例工作区，但大多数用户将创建他们自己的工作区。在这一节中，假定您将使用示例工作区或您自己使用 **FME Workbench** 创建的工作区。

### 如何列出存储库中的工作区

工作区名称可以使用存储库方法中的获取工作区概要方法快速列出。此外，可使用其他存储库方法读取有关工作区的更多详细信息。

```
IFMERepository[] repositoryList = repositoryMgr_.GetRepositories(null);
foreach (IFMERepository repository in repositoryList)
{
    IFMEWorkspaceSummary[] workspaceSummaries = repository.GetWorkspaceSummaries(null);
    foreach (IFMEWorkspaceSummary workspaceSummary in workspaceSummaries)
    {
        Console.WriteLine("Repository:{0} {1} {2} ON:{3}\n",
            repository.Name,
            workspaceSummary.Title,
            workspaceSummary.Name,
            workspaceSummary.IsEnabled);
    }
}
```

### 如何将工作区添加到存储库中

要将一个新工作区添加到存储库，存储库名称、工作区文件路径（如 "C:\myworkspaces\foo.fmw"）和工作区名称（如 foo.fmw）是必需的。在添加一个工作区时，实际的工作区文件上传到了 **FME Server** 存储库中。

```
IFMERepository repository = repositoryMgr_.GetRepository(repositoryName);
if (!repository.ItemExists(workspaceName))
{
    repository.AddItem(workspaceName, workspaceFilePath);
}
```



```

else
{
    throw new FMEServerException("There is already a workspace
with this name!");
}

```

## 如何更新存储库中的工作区

要更新存储库中的工作区，存储库名称、工作区文件路径（如 "C:\myworkspaces\foo.fmw"）和工作区名称（如 foo.fmw）是必需的。由于工作区名称唯一标识了存储库中的工作区，因而需要与您想要更新的工作区匹配。当更新工作区时，实际的工作区文件上传到了 **FME Server** 存储库中。

```

IFMERepository repository =
repositoryMgr_.GetRepository(repositoryName);
if (repository.ItemExists(workspaceName))
{
    repository.UpdateItem(workspaceName, workspaceFilePath)
}
else
{
    throw new FMEServerException("There is no workspace with
this file name!");
}

```

## 如何从存储库中获得工作区

要从存储库中获得一个工作区，存储库名称、本地工作区文件路径（如 "C:\myworkspaces\foo.fmw"）和工作区名称（如 foo.fmw）是必需的。由于工作区名称唯一标识了存储库中的工作区，因而需要与您想要获取的工作区匹配。获取一个工作区时，实际工作区文件从 **FME Server** 存储库中下载到指定的本地工作区文件路径下。

```

IFMERepository repository =
repositoryMgr_.GetRepository(repositoryName);
if (repository.ItemExists(workspaceName))
{
    repository.GetItem(workspaceName, workspaceFilePath);
}
else
{
    throw new FMEServerException("There is no workspace with

```



```
this file name!");  
}
```

## 如何从存储库中删除工作区

在存储库中删除一个工作区，需要存储库名称和工作区的名称。

```
IFMERepository repository =  
repositoryMgr_.GetRepository(repositoryName);  
if (!repository.RemoveItem(workspaceName))  
{  
    throw new FMEServerException("There is no workspace with  
this file name!");  
}
```

## 使用资源

一些工作区要成功运行的话可能需要资源。 与工作区相关联的任何资源都存放在工作区中，并可通过存储库对象来访问。

## 如何列出工作区的资源

若要查看已存在的工作区的资源，需要存储库名称和工作区的名称。

```
IFMERepository repository =  
repositoryMgr_.GetRepository(repositoryName);  
IFMEWorkspace workspace =  
repository.GetWorkspace(workspaceName);  
  
IFMEResource[] resources = workspace.Resources;  
foreach (IFMEResource resource in resources)  
{  
    Console.WriteLine("{0} {1}\n", resource.Name,  
resource.Description);  
}
```

## 如何添加资源到工作区中

要为工作区添加新的资源，存储库名称、工作区名称， 本地资源文件路径（如“C:\myresources\resource.csv”） 和资源名称（如 resource.csv）是必需的。在添加资源时，实际资源文件上传到了 FME Server 存储库中。

```

IFMERepository repository =
repositoryMgr_.GetRepository(repositoryName);
if (!repository.ResourceExists(workspaceName,
resourceName))
{
    repository.AddResource(workspaceName, resourceName,
resourceFilePath);
}
else
{
    throw new FMEServerException("There is already a resource
with this name!");
}

```

## 如何更新工作区中的资源

若要更新工作区的资源，存储库名称、工作区名称，本地资源文件路径（如“C:\myresources\resource.csv”）和资源名称（如 resource.csv）是必需的。由于资源名称唯一标识了工作区中的某个资源，因而需要与您想要更新的资源相匹配。更新资源时，实际资源文件上传到了 **FME Server** 存储库中。

```

IFMERepository repository =
repositoryMgr_.GetRepository(repositoryName);
if (repository.ResourceExists(workspaceName, resourceName))
{
    repository.UpdateResource(workspaceName, resourceName,
resourceFilePath);
}
else
{
    throw new FMEServerException("There is no resource with
this name!");
}

```

## 如何为工作区获取资源

若要为工作区获取资源，存储库名称、工作区名称，本地资源文件路径（如“C:\myresources\resource.csv”）和资源名称（如 resource.csv）是必需的。由于资源名称唯一标识了工作区中的某个资源，因而需要与您想要获取的资源相匹配。获取资源时，实际资源文件从 **FME Server** 存储库下载到本地资源文件路径。



```

IFMERepository repository =
repositoryMgr_.GetRepository(repositoryName);
if (repository.ResourceExists(workspaceName, resourceName))
{
    repository.GetResource(workspaceName, resourceName,
resourceFilePath);
}
else
{
    throw new FMEServerException("There is no resource with
this name!");
}

```

## 如何为工作区删除资源

从工作区中删除特定的资源，需要存储库名称、工作区名称和资源名称。删除特定的工作区将删除所有工作区中相关的资源。

```

IFMERepository repository =
repositoryMgr_.GetRepository(repositoryName);
if (!repository.RemoveResource(workspaceName,
resourceName))
{
    throw new FMEServerException("There is no resource with
this name!");
}

```

## 使用服务

**FME Server** 的默认安装了多项服务，每个服务都为最终用户提供一个特定的功能。服务对象为您提供了有关这些服务的信息，如服务名称和服务 **URL** 模式。它还允许为您可能希望创建的服务创建新的服务描述。

## 如何列出可用的服务

若要列出可用的服务，需要存储库管理器对象。

```

IFMEService[] services = repositoryMgr_.GetServices(null);

foreach (IFMEService service in services)
{
    Console.WriteLine("{0} {1} {2} ON:{3} {4}\n",

```

```
        service.DisplayName,  
        service.Name,  
        service.URLPattern,  
        service.IsEnabled,  
        service.Description);  
    }
```

## 如何添加服务

添加一项新服务、需要服务名称、 服务显示名，描述和 **URL** 模式。

```
IFMEService service = serverSession_.CreateService(name,  
displayName, description, urlPattern);  
  
if (!repositoryMgr_.ServiceExists(name))  
{  
    repositoryMgr_.AddService(service);  
}  
else  
{  
    throw new FMEServerException("There is already a service  
with this name!");  
}
```

## 如何更新服务

更新一项服务，需要服务名称、 服务显示名、描述和 **URL** 模式。 服务名称唯一地标识一个服务。

```
IFMEService service = serverSession_.CreateService(name,  
displayName, description, urlPattern);  
  
if (repositoryMgr_.ServiceExists(name))  
{  
    repositoryMgr_.UpdateService(service);  
}  
else  
{  
    throw new FMEServerException("There is no service with  
this name!");  
}
```

## 如何删除服务



要删除一个服务，只需要唯一标识该服务的服务名称即可。

```
if (!repositoryMgr_.RemoveService(name))
{
    throw new FMEServerException("There is no service with
this name!");
}
```

## 运行工作区

一旦工作区发布到一个 **FME Server** 存储库之后，便可以运行该工作区。这可以通过构建并提交使用转换管理器对象的任务请求的方式来完成。用户通常会创建他们自己的工作区，但此示例中，我们将使用 **FME Server** 默认安装包包含的示例工作区。

第一步是连接到转换管理器，**FME Server** 的特定组件将处理任务请求。

```
IFMETransformationManager transformationMgr =
session_.GetTransformationManager();
```

下一步，我们需要创建将提交到 **FME Server** 的转换请求。预期的参数有：

- **FME Server** 节点分支名称，在节点的配置中定义的。  
**SERVER\_CONSOLE\_CLIENT** 是一个这种分支，附带了 **FME Server** 的默认配置的一部分。每分支指定各种处理指令（例如输出放置的位置文件夹）。不同分支的信息请参阅 **< FMEServer InstallDir >**  
**\Server\fmeServerConfig.txt**
- 包含目标文件的存储库唯一名称，**demodownload** 是一个此类存储库，它包含 **"austin.fmw"** 的工作区。
- 工作区、映射文件或 **FME** 要执行的命令文件名。在后者的情况下，假设该文件驻留在已设置为我们选择的第一个参数的分支 **FME\_MAPPING\_DIR**。在这里，我们指定从 **demodownload** 存储库中的工作区调用 **austin.fmw**。

```
IFMETransformationRequest req =
session_.CreateTransformationRequest("SERVER_CONSOLE_CLIENT",
"demodownload", "austin.fmw");
```



一些工作区发布了可以设置的参数。一些具有默认值，但其他的参数需要在我们能够运行工作区之前显式设置。这取决于工作区自身。若要设置此类已发布的参数——在此示例中是 **MAX\_FEATURES**:

```
req.SetPublishedParameter("MAX_FEATURES", "2000");
```

**注意:** 已发布的参数的值总是必须作为字符串传递而不管他们实际上代表何种类型。

**FME Server** 中也存在始终可用的指令。要获取这些指令的完整列表，请参阅 **FME Server** 文档。对于我们的示例，我们会将该任务的优先级设置为较高的值，以通知 **FME** 服务器在其他任务之前执行该任务。这不是必要的，但之所以这样做只是用来说明如何设置指令。

```
req.SetTMDirective("priority", 42);
```

一旦设置好了所有的参数和指令，我们可以将请求提交到 **FME Server**，请求结果存储在转换结果对象中。

```
IFMETransformationResult result =  
transformer.TransactJob(req);
```

转换结果对象有一系列方法从 **FME Server** 响应中获取信息的不同部分。在我们的示例中，我们首先显示从服务器接收的原始的、未解析的字符串。在某些特定情况下，客户端可能要做自己对此字符串解析，而不是依赖提供方法来解析。

```
Console.WriteLine(result.FMEServerResponse);
```

我们现在可以显示更多的信息。首先，我们检查转换是否成功，如果成功，我们列出一些返回的属性。属性存储为 “名称 = 值” (*name = value*) 对，虽然有些可能为 **null**。

```
if (!result.TransformationSuccess)  
{  
    throw new FMEServerException("Transformation  
failed.\nThe error was: " + result.StatusMessage);  
}  
else  
{  
    Console.WriteLine("Transformation successful!");  
}
```



```
    Console.WriteLine("The following information was parsed  
from the response:\n");  
    Console.WriteLine("=====");  
    foreach (KeyValuePair pair in result.GetAllProperties())  
    {  
        Console.WriteLine(pair.Key + " = " + pair.Value);  
    }  
    Console.WriteLine("=====");  
}
```

## 开发者资源

要了解 FME Server 应用开发的概况, 请参阅 [FME Server Administrator's Guide](#), Appendix C.

### FME Server APIs

- [Samples \(C++, Java, dotNet\)](#)
- [C++ documentation](#)
- [Java documentation](#)
- [dotNet documentation](#)
- [SOAP documentation](#)

### FME Server Web Services API

- [FME Server Web Services API Reference](#)